List decoding: Algorithms and Applications

Madhu Sudan

MIT

Joint works with

– Sanjeev Arora
– Venkatesan Guruswami
– Luca Trevisan & Salil Vadhan.

This talk

- The list decoding problem for "polynomial codes".
- Application to hardness amplification.
- Algorithm for univariate case.
- Algorithm for multivariate case.

Error-Correcting Codes

Defn: (Linear) Error-correcting code C
over q-ary alphabet $\mathbb{F}_q$ is
a k-dimensional subspace of $\mathbb{F}_q^n$.

- Block length$(C) = n$.
- Information content$(C) = k$.
- Distance$(C) = \delta \stackrel{\text{def}}{=}$ min Hamming distance between two vectors in C.

Abbreviation: C is an $[n, k, \delta]_q$ code.

Multivariate Polynomial codes

Lemma: [DeMillo+Lipton,Schwartz,Zippel]
Two degree d polynomials over $\mathbb{F}_q$
agree on $\leq d/q$ fraction of inputs.

$\Rightarrow$ m-variate degree d polynomials over $\mathbb{F}_q$
define an $[n,k,\delta]_q$ code $C_{m,d,q}$, where:

– $n = q^m$.
– $k = \binom{m+d}{m}$.
– $\delta = n(1 - \frac{d}{q})$.

Note: exponential dependence on m.

- $m = 1$: Reed-Solomon codes.
- $d = 1$: Subfamily of Hadamard codes.
- Arbit: Reed-Muller codes.

List Decoding Problem

Defn: [Elias '57-'88]

Given: $[n, k, \delta]_q$ code $C_{m,d,q}$,
a vector $f \in \mathbb{F}_q^n$
error parameter e.

Output: All codewords of C
within distance e from f.

- $e < \delta/2$: Classical error-correction.
- Our goal: $e \to \delta$!
- Desired running time: $\mathrm{poly}(m, d, q)$?

$\Omega(n) = \exp(m)$ is a lower bound - right?

Introducing: Oracles, Randomness ...

- Codewords, f: Functions mapping $\mathbb{F}_q^m \rightarrow \mathbb{F}_q$.
- Represent them as oracles (strings with random access).
  - Input ($= f$)= Oracle.
  - Output = (Randomized) Oracle machines invoking f as subroutine.
- Allow randomized algorithms.
- Motivated by notions in Program Checking, PCPs.
- First use? [Goldreich-Levin]?

Results

Thm. 1: $\exists$ prob. list decoding algorithm
for every $C_{m,d,q}$
Running time $\mathrm{poly}(m, d, \log q)$
provided $e < n(1 - O(\sqrt{d/q}))$.

(If $\delta = n - o(n)$, then $e = n - o(n)$.)

Thm. 2: $\forall\ k, \epsilon, q,$ $\exists$ an $[n, k, \delta]_q$ code:
$n = \left(\frac{k \cdot q}{\epsilon}\right)^{O(1)}$.
list decoding time $\mathrm{poly}(\log n, \frac{1}{\epsilon})$,
provided $e \le n\left(1 - \frac{1}{q} - \epsilon\right)$.

Thm 1 $\Rightarrow$ Thm 2: Use $C_{m,d,q^l} + C_{l,1,q}$
for appropriate choice of m, d, l.
(list-decoded using Thm 1 + brute force).

Application to hardness amplification

Given: $g : \{0,1\}^l \rightarrow \{0,1\}$ hard on worst-case
Want: h hard on average.

[NW '94]: Existence of h in E
hard on average for subexp. circuits
implies $\mathrm{BPP} = \mathrm{P}$.

[IW '97]: Construct h from g in E
hard on worst-case for subexp. circuits.

Alternate construction

- $C = [n, 2^l, \delta]_2$ (systematic) code (from Thm 2.)
- $g = 2^l$-bit string to be encoded.
- Then $h = C(g)$ is hard on average.

Analysis:

- Let f predict h with accuracy $\frac{1}{2} + \epsilon$.
- Let $f_1, \ldots, f_k$ be outputs of list decoder.
- Some f_i's is $h = C(g)$; hence computes g efficiently.

Algorithms for list decoding

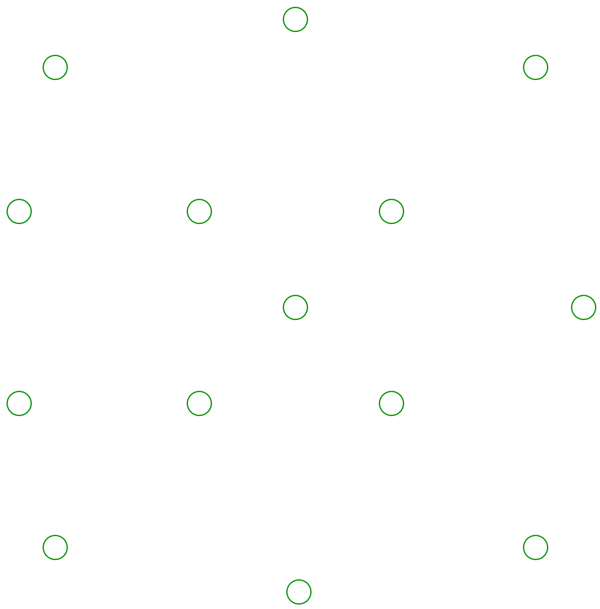
- Univariate Polynomials
  - Solution using algebraic methods. (Factorization of bivariate polynomials.)
  - [S. '97], [Guruswami+S. '98]. Initial work [Ar,Lipton,Rubinfeld,S.'92].
  - Note "oracles", random-access etc. unimportant here.
- Multivariate Polynomials
  - Solution by probabilistic reduction to univariate case.
  - [Arora+S. '97], [S+Trevisan+Vadhan '99].

Problem Statement Rephrased

Univariate version:

Input: $\{(x_1, y_1) \ldots, (x_n, y_n)\}$: $x_i, y_i \in \mathbb{F}_q$.
Parameters t, d;
Output: All polynomials p
of degree at most d
s.t. $p(x_i) = y_i$ for at least t values of i.

An Example

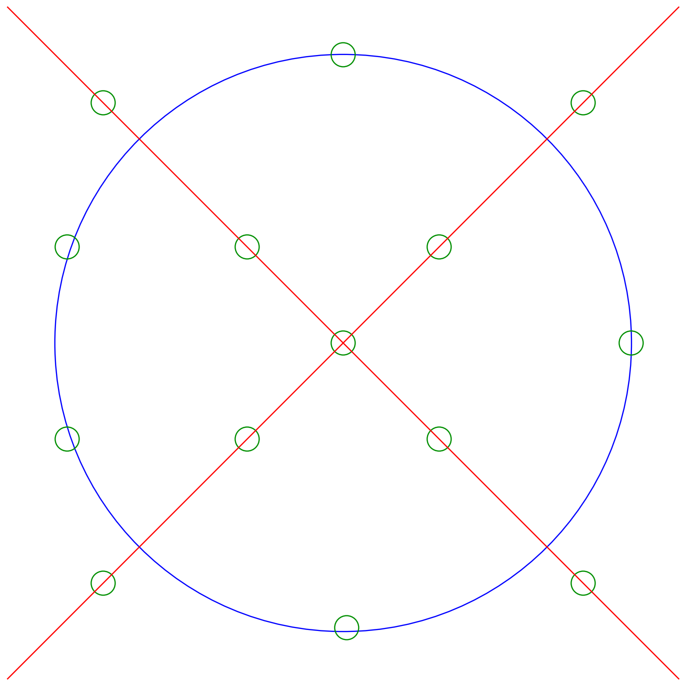


$$n = 14\ ; \qquad d = 1\ ; \qquad t = 5$$

Solution Idea

- Explain relation (x_i, y_i) algebraically.
- I.e. Find non-zero $Q(x, y)$ such that $Q(x_i, y_i) = 0$ for every i. (Ideally degree of Q small in both variables.)
- Hopefully p emerges in picture!
- Formally $(y - p(x) \mid Q(x, y)$.

A Quartic Fit: All the Lines Emerge



$$Q(x, y) = y^4 - x^4 + y^2 - x^2$$

After factoring

$$Q(x, y) = (y^2 + x^2 - 1)(y + x)(y - x)$$

Conditions for divisibility of Q

- $\deg_x(Q), \deg_y(Q)$ small.
- $\deg(p)$ small.
- # pts. s.t. $Q(x_i, y_i) = y_i - p(x_i) = 0$ large.

Lemma: $\deg_x(Q), \deg_y(Q) \leq D$, $\deg(p) \leq d$, $t \geq D(d+1)$ implies $y - p(x)$ divides $Q(x, y)$.

Proof? Similar to the fact that two degree k polynomials can not agree in $k+1$ places.

Warning: $Q \not\equiv 0$.

Finding $Q(x, y)$ with low degree

- Q1: How to find Q of low degree?
- Q2: Why does it even exist?

- A1: Solve linear system. Let $Q(x, y) = \sum_{j,l} q_{jl} x^j y^l$.

  Unknowns: $\{q_{jl} : 0 \leq j, l \leq D\}$.
  Constraints: $\sum_{j,l} q_{jl} \ \ x_i^j \ \ y_i^l \ = \ 0; \ \ i \ \in \{1, \dots, n\}$.

- A2: Solution exists if

$$\# \text{ unknowns} \quad > \quad \# \text{ constraints}$$
$$(D + 1)^2 > n$$

Summary

Algorithm:

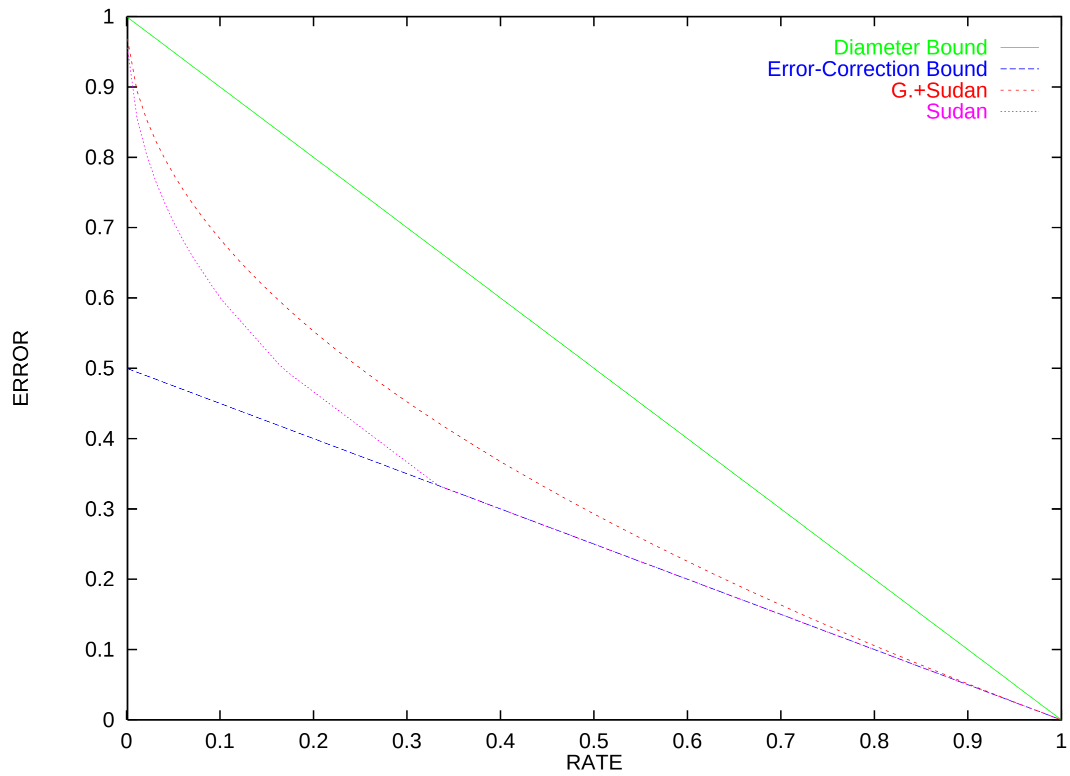
Step 1: Find $Q \not\equiv 0$ of degree $\sqrt{n}$ s.t. $Q(x_i, y_i) = 0$ for every $i \in \{1, \ldots, n\}$.

Step 2: Factor $Q(x, y)$ and report all p such that $y - p(x)$ divides Q.

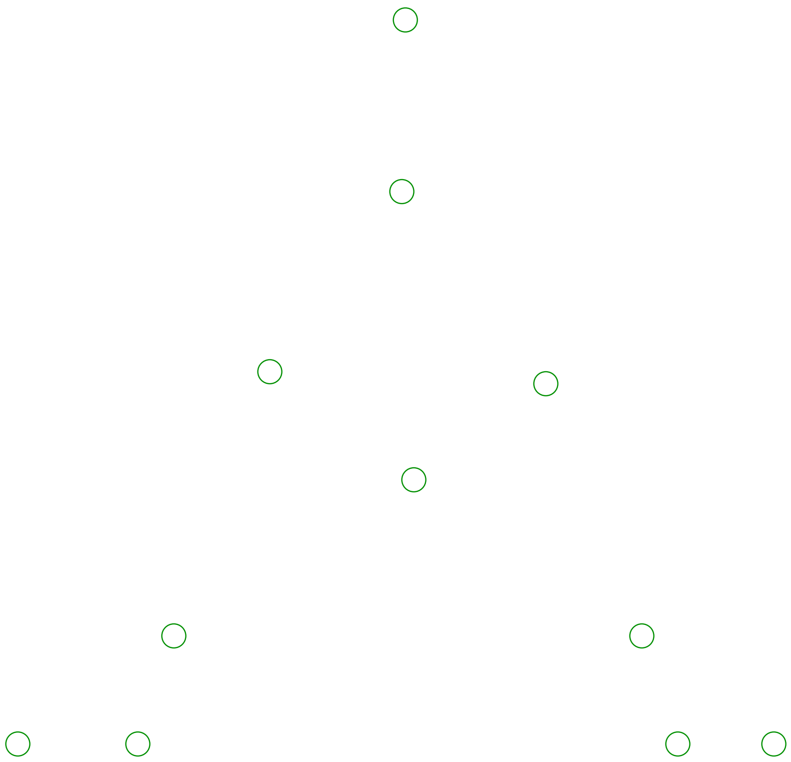
Theorem: Can find every p with agreement t, provided $t \geq (d+1)\sqrt{n}$.

Improvements:
– Careful choice of $Q \Rightarrow t \geq \sqrt{2dn}$.
Further Improvements:
Improved algorithm $\Rightarrow t \geq \sqrt{dn}$.

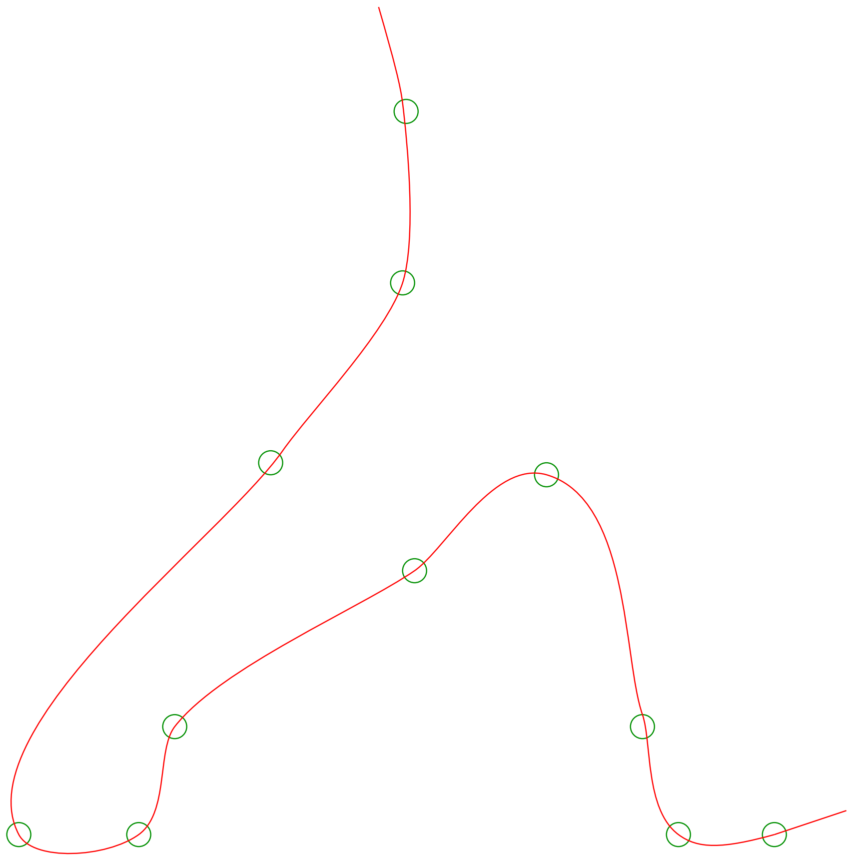
Algorithms & Their Performance



Another Example



A Quartic Fit: No information!!



Idea behind handling this

- Similar Plan: Fit Q, Hope for factors of form $y - p(x)$ for all "relevant" p.
- Main Difference: Expect more from Q.
  - Every (x_i, y_i) is a "singularity" of Q. (Expressible as 3 linear constraints on coefficients of Q.)

Why this will help?

What we lose? – Additional constraints force us to raise the degree of Q. (We need more unknowns to guarantee existence of non-trivial solution.)

But....

Much more gain in the second phase when we factor Q and look for factors of the form $y - p(x)$!

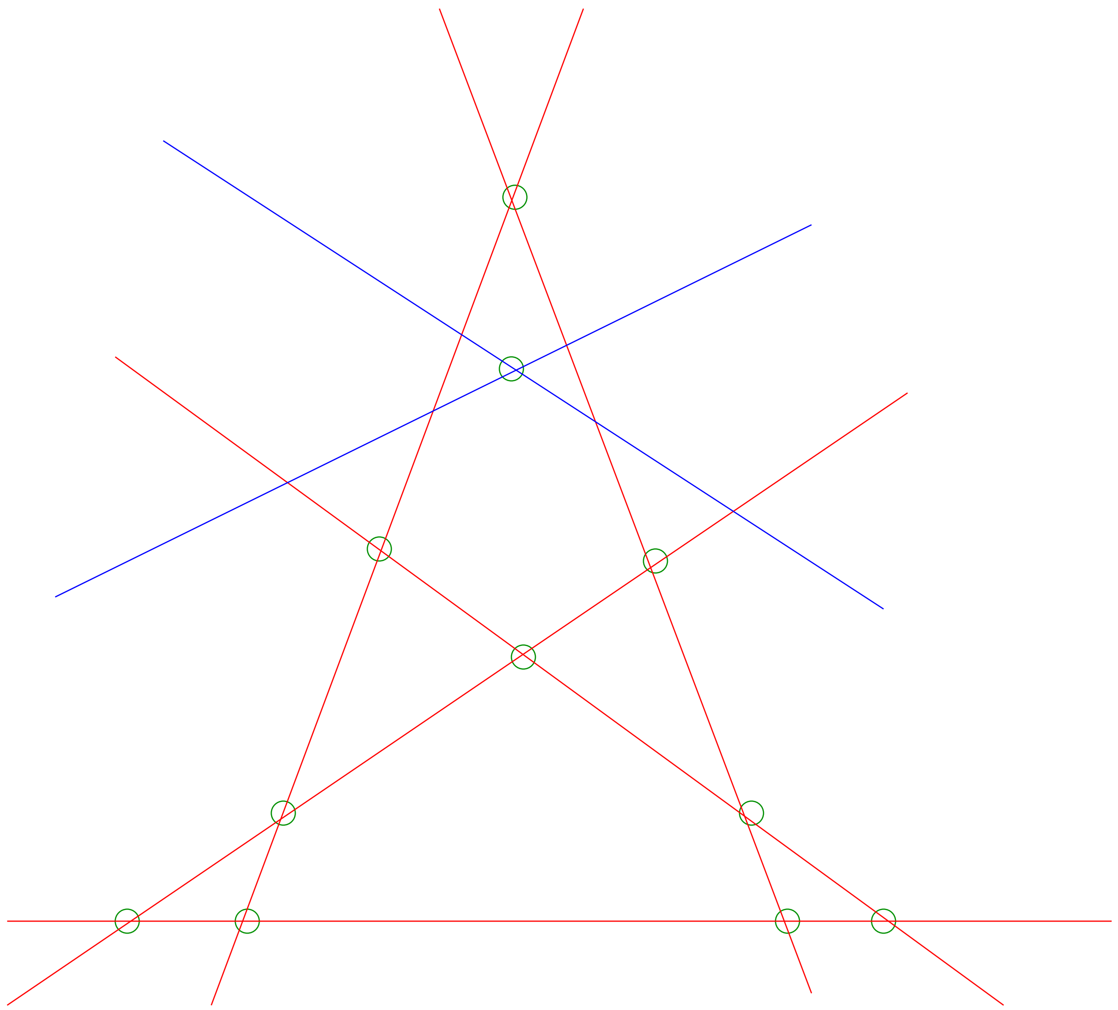
- Now p passes through singularities of Q as opposed to just points on Q.
- Need only half as many singularities as regular points!!

Back to the Example

Fit $Q(x, y)$ of total degree 7 through the points s.t Q "intersects itself" at each of the 11 points. (Can do this: solve homogeneous linear system with $\binom{7+2}{2} = 36$ unknowns and $11 \times 3 = 33$ constraints.)

Now all relevant linear polynomials "emerge" in the picture!!

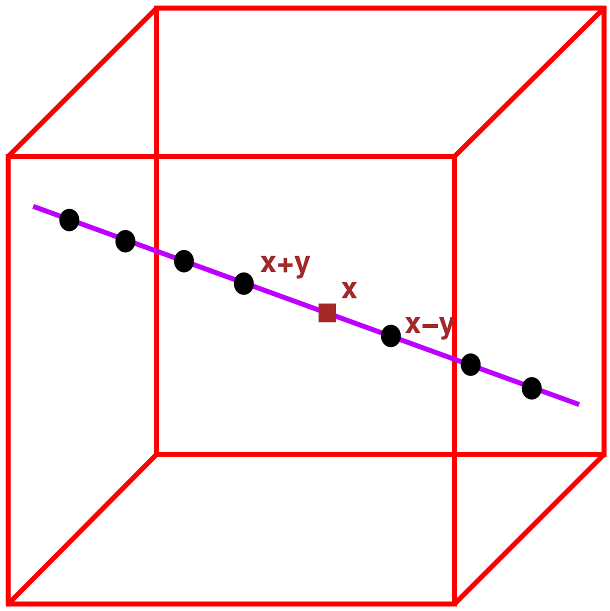
Degree 7 Super-fit: All Lines Emerge!!



Multivariate Problem Statement

Input: Oracle $f : \mathbb{F}_q^m \longrightarrow \mathbb{F}_q$ and d, ρ.
Output: An oracle for every degree d poly p
s.t. p and f agree in ρ fraction of inputs.

Solution: Reduction to univariate case.



– Look at random line in $\mathbb{F}_q^m$.
– For $x, y \in \mathbb{F}_q^m$, $l = \{l(t) = x + ty | t \in \mathbb{F}_q\}$.
– For $f : \mathbb{F}_q^m \longrightarrow \mathbb{F}_q$, let $f|_l(t) = f(l(t))$.

Lines in m dimensional space

- Algebraic properties:
  - If $p : \mathbb{F}_q^m \to \mathbb{F}_q$ is a poly of total degree d then $p|_l$ is a univ. poly of degree d.
- Randomness properties:
  - For random x, y, l is a collection of pair-wise independent samples of $\mathbb{F}_q^m$.
  - W.h.p. $f|_l$ and $p|_l$ have nearly same agreement on l as on whole space $\mathbb{F}_q^m$. (Chebychev Inequality)
  - Even if x fixed, w.h.p. over y agreement of $f|_l$ and $p|_l$ not too low. (Markov Inequality)
- Used in [BF,GLRSW,GS] to get "self-corrector" ($\rho > \frac{1}{2}$).

First cut

Given: Oracle for $f : \mathbb{F}_q^m \to \mathbb{F}_q$; and $x \in \mathbb{F}_q^m$.

(Fix poly p ρ-agreeing with f.)

Output: $p(x)$.

Algorithm:

– Pick $y \in_R \mathbb{F}_q^m$; let $l = \{l(t) = x + ty\}$.
– Call univariate decoding:

  $\{(t, f|_l(t)) | t \in \mathbb{F}_q - \{0\}\}$;
  with parameters $\rho' < \rho$, d.
– $\rho' \ll \rho \Rightarrow f|_l$ ρ'-agrees with $p|_l$.
– $\Rightarrow$ Output of univ. alg. includes $p|_l$.
– Which one is p?

When $\rho < \frac{1}{2}$

Main Issue: Want to compute $p(x)$;
but don't know p!

Only "information" about p is f.

Suppose we know: $p(z) = a$
z of algorithm's (random) choice.
Suffices to determine p uniquely (w.h.p.)!

Algorithm

$A_{z,a}(x)$:

- Let $l(t) = x + t(z - x)$ and $l = \{l(t)\}$.
- Call univ. decoding with $\{t, f|_l(t))\}$; ρ', d.
- Let $g_1, \ldots, g_k$ be output of univ. alg.
- If $\exists$! poly g_j with $g_j(1) = a$,
  then output $g_j(0)$ else error.

(Deterministic! Always uses z.)

Initialization: Pick z and random.
For every $a \in \mathbb{F}_q$ output
"Self-correct"$(A_{z,a})$.

Analysis

Definition: (x, z) bad if:

1. $p|_l$ and $f|_l$ do not have ρ' agreement. OR
2. Some (wrong) output $g_{j'}(z) = p|_l(z)$.

Claim: (x, z) not bad implies output $= p(x)$.

Claim: 1. happens with very low probability.
Proof: Chebychev's Inequality.

Claim: 2. happens w.p. $\leq (k-1)d/|F|$.
Proof:

- Choose line first and z later.
- Let g_j be correct poly ($p|_l$).
- z bad if $\exists\, j' \neq j$ s.t. $g_j(z) = g_{j'}(z)$.
- $\#$ choices of j' $k-1$.
- $\Pr[\text{z bad for } j'] \leq d/|F|$.

Analysis-II

$$\tau \stackrel{\mathrm{def}}{=} \Pr_{x,z}[1. \cup 2.]$$

$\Pr_z[$ For $\sqrt{\tau}$ fraction of x's, (x, z) bad $] \leq \sqrt{\tau}$.

Thus, with probability $1-\sqrt{\tau}$ (over the choice of z) $A_{z,p(z)}$ computes p correctly on $1 - \sqrt{\tau}$ fraction of inputs (x).

Thm. 2: List decoder for $C_{m,d,q}$ recovering all polynomials with ρ agreement with run time poly(m, d, q) provided $q = \Omega(d/\rho)^{O(1)}$.

Conclusions

- Further examination of list decoding problem.
- Combinatorics?
- Algorithms?
- Hardness?
- Applications?